Low-Rank Update Eigensolver for Supercell Band Structure Calculations

Ming Gu*, Jianlin Xia**, and David Z.-Y. Ting†

*, ** Dept. of Mathematics, Univ. California at Berkeley, Berkeley CA 94720-3840 USA.
* [mgu@math.berkeley.edu]

† Jet Propulsion Laboratory, California Institute of Technology,
Pasadena, CA 91109-8099 USA. [David.Z.Ting@jpl.nasa.gov]

In supercell electronic or photonic band structure calculations we wish to find energy or frequency normal modes as functions of wave vectors (**k**) for two- or three-dimensional periodic structures. A straightforward approach involves the application of Bloch's Theorem to limit the real-space computation domain to a single supercell, and discretizing Schrödinger or Maxwell equations using a finite differencing scheme to obtain a matrix representing the problem. A key characteristic of the matrix is that only the elements involving the discretized points on the surface of the supercell are **k**-dependent. So in varying **k**, only a small fraction (essentially ratio between the number of dicretized surface points and number volume points) of the matrix elements are updated. The low-rank update eigensolver exploits this property, and allows us to accelerate band structure calculations by solving the eigenvalue problem once, and then treat the remaining cases much more rapidly. As an example, Table 1 shows timing results on the application of the low-rank update method to the computation of band structures of a 2D photonic crystal using grid sizes of 16x16, 32x32, and 64x64. It shows that the amount of time required for low rank updates is significantly less than the initial set up cost. Also, the benefit increases considerably with increasing grid size, as a consequence of decreasing "surface-to-volume" ratio.

Table 1 Timing comparison between Low-rank update method and Lapack eigensolver.

Matrix Size (Rank 2)	Low-Rank Update Method		Lapack Eigensolver Time per case (sec.)
	Time for computing the first case (sec.)	Time for computing each subsequent case (sec.)	
256	1.5	0.48	0.4
1024	39.5	3.2	55.6
4096	3664.8	23.2	15241.0